UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/828,271	04/05/2001	Vijayan Rajan	5693p286	6350

48102 7590 04/14/2009
NETWORK APPLIANCE/BSTZ
BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP
1279 OAKMEAD PARKWAY
SUNNYVALE, CA 94085-4040

EXAMINER	
ZHEN, LI B	

ART UNIT	PAPER NUMBER
2194	

MAIL DATE	DELIVERY MODE
04/14/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	**Application No.** 09/828,271	**Applicant(s)** RAJAN ET AL.	
	Examiner LI B. ZHEN	**Art Unit** 2194	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *11 December 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *11,12,14,16,23 and 30-37* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *11,12,14,16,23 and 30-37* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

DETAILED ACTION

1. Claims 11, 12, 14, 16, 23 and 30 – 37 are pending.

Response to Arguments

2. Applicant's arguments with respect to the claims have been considered but are moot in view of the new ground(s) of rejection.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. **Claims 11, 12, 14, 16, 23 and 30 – 37 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Parallel Job Scheduling: Issues and Approaches" [hereinafter Feitelson] in view of U.S. Patent No. 6,745,222 to Jones et al. [hereinafter Jones].**

5. As to claim 11, Feitelson teaches a method comprising:

running a plurality of tasks [threads; pp. 6 – 7, Section 3.1] in a multiprocessor system that includes a plurality of processors [processors; pp. 6 – 7, Section 3.1];

scheduling the plurality of tasks using a plurality of scheduling domains [different jobs execute in different protection domains; p. 5, Section 2.3 and p. 6, Section 3] by scheduling tasks on a processor independent of the identity of the processor [Global Queue scheduling; Section 3.1, p. 7], wherein none of the plurality of scheduling domains is bound to any one processor of the plurality of processors [Dynamic Partitioning, Section 3.3, p. 9];

wherein each of the one or more resources is assigned to one of the scheduling domains [two-level scheduling scheme; pp. 9 – 10] and is not subject to domain migration [Section 3.2 Variable Partitioning, p. 8], wherein implicitly synchronizing the tasks comprises prohibiting tasks that are each associated with a same scheduling domain from running concurrently [large partition can be divided into smaller partitions to allow several small jobs to execute in parallel; Section 3.2, p. 8], and

allowing tasks that are each associated with different scheduling domains to run concurrently [small jobs to execute in parallel; Section 3.2, p. 8]; and

changing association of a task of the plurality of tasks from a first scheduling domain to a second scheduling domain, if the task requires a shared resource assigned to the second scheduling domain [thread migration; p. 7, 3rd paragraph]. Feitelson does not specifically teach implicitly synchronizing the tasks with regard to one or more resources shared by the tasks in the system by associating the tasks with the scheduling domains.

However, Jones teaches running a plurality of tasks [col. 24, lines 15 – 52] in a multiprocessor system that includes a plurality of processors [col. 7, line 63 – col. 8, line 13], each processor having an identity [col. 22, lines 28 – 38], scheduling the plurality of tasks using a plurality of scheduling domains [group of related threads; col. 23, lines 15

- 30], implicitly synchronizing the tasks with regard to one or more resources shared by the tasks in the system by associating the tasks with the scheduling domains [Synchronization mechanisms are a feature of multitasking operating systems that coordinate the use of a particular resource of the computer system by different threads; col. 28, lines 6 – 24], prohibiting tasks that are each associated with a same scheduling domain from running concurrently ["blocks" or suspends the execution of, the thread attempting to acquire the mutex, until the mutex becomes available; col. 28, lines 6 – 25] and allowing tasks that are each associated with different scheduling domains to run concurrently [threads in the gang are attempted to be scheduled on different processors at the same times, allowing them to make progress in parallel as a group; col. 23, lines 15 – 31].

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to modify the invention of Feitelson to incorporate the features of Jones. One of ordinary skill in the art would have been motivated to make the combination because this provides predictable scheduling of real-time programs and non-real-time programs using a repeating precomputed schedule [col. 3, lines 19 – 52 of Jones].

6. As to claim 12, Feitelson as modified teaches a system comprising:

a plurality of processors [processors; pp. 6 – 7, Section 3.1 of Feitelson and col. 7, line 63 – col. 8, line 13 of Jones], each processor having an identity [col. 22, lines 28 – 38 of Jones];

a memory coupled to each of the plurality of processors [col. 7, lines 32 – 63 of Jones], the memory storing data defining a set of tasks [threads; pp. 6 – 7, Section 3.1 of Feitelson and col. 24, lines 15 – 52 of Jones], each task of the set of tasks being runnable on more than one of the processors [p. 5, Section 2.3 and p. 6, Section 3 of Feitelson], each the task being associated with one of a plurality of scheduling domains [Section 3.1, p. 7 of Feitelson], each of the plurality of scheduling domains controlling one or more shared resources [pp. 9 – 10 of Feitelson], wherein each of the shared resources is not subject to domain migration [Section 3.2 Variable Partitioning, p. 8 of Feitelson]; and

a scheduler to schedule the set of tasks using a plurality of scheduling domains [p. 5, Section 2.3 and p. 6, Section 3 of Feitelson] by scheduling tasks on a processor independent of the identity of the processor [Section 3.1, p. 7 of Feitelson], wherein none of the plurality of scheduling domains is bound to any one processor of the plurality of processors [Section 3.3, p. 9 of Feitelson], where the scheduler prohibits tasks that are each associated with a same scheduling domain from running concurrently [Section 3.2, p. 8 of Feitelson and col. 28, lines 6 – 25 of Jones] but allows tasks that are each associated with a different one of the plurality of scheduling domains to run concurrently [small jobs to execute in parallel; Section 3.2, p. 8 of Feitelson and col. 23, lines 15 – 31 of Jones], and wherein the scheduler changes association of a task of the set of tasks from a first scheduling domain to a second scheduling domain if the task requires a shared resource controlled by the second scheduling domain [thread migration; p. 7, 3rd paragraph of Feitelson].

7. As to claim 23, Feitelson as modified teaches a process comprising:

scheduling a plurality of tasks in a multiprocessor system that includes a plurality of processors [processors; pp. 6 – 7, Section 3.1 of Feitelson and col. 7, line 63 – col. 8, line 13 of Jones], each processor having an identity [col. 22, lines 28 – 38 of Jones], by scheduling tasks on a processor independent of the identity of the processor [Section 3.1, p. 7 of Feitelson and col. 24, lines 15 – 52 of Jones], wherein none of the plurality of scheduling domains is bound to any one processor of the plurality of processors [Section 3.3, p. 9 of Feitelson];

performing implicit synchronization of the plurality of tasks [col. 28, lines 6 – 24 of Jones], the implicit synchronization dividing the tasks into the scheduling domains [Section 3.2, p. 8 of Feitelson], at least one of the scheduling domains being associated with at least two tasks of the plurality of tasks [pp. 6 – 7, Section 3.1 of Feitelson and col. 24, lines 15 – 52 of Jones], wherein each of the shared resources is not subject to domain migration [Section 3.2 Variable Partitioning, p. 8 of Feitelson], and a resource shared by the at least two tasks [pp. 9 – 10 of Feitelson], and wherein tasks within a same scheduling domain are prohibited from running concurrently even if run on different processors [Section 3.2, p. 8 of Feitelson and col. 28, lines 6 – 25 of Jones] and tasks that are each from a different scheduling domain are allowed to run concurrently [small jobs to execute in parallel; Section 3.2, p. 8 of Feitelson and col. 23, lines 15 – 31 of Jones]; and moving a task of the plurality of tasks from a first scheduling domain to a second scheduling domain, if the task requires a resource controlled by the second scheduling domain [thread migration; p. 7, 3rd paragraph of Feitelson].

8. As to claim 30, Feitelson as modified teaches a method of scheduling a plurality of processes in a multiprocessor system, the method comprising:

associating the plurality of processes [pp. 6 – 7, Section 3.1 of Feitelson and col. 24, lines 15 – 52 of Jones] with a plurality of scheduling domains [p. 5, Section 2.3 and p. 6, Section 3 of Feitelson] wherein none of the plurality of scheduling domains is bound to any one processor in the system [Section 3.3, p. 9 of Feitelson], and wherein each of the processes is executed by a processor independent of an identity of the processor [Section 3.1, p. 7 of Feitelson];

implicitly synchronizing the plurality of processes with regard to one or more shared resources [col. 28, lines 6 – 24 of Jones] by prohibiting concurrently executing processes that are each associated with a same scheduling domain [Section 3.2, p. 8 of Feitelson and col. 28, lines 6 – 25 of Jones] but allowing concurrently executing processes that are each associated with a different one of the plurality of scheduling domains [small jobs to execute in parallel; Section 3.2, p. 8 of Feitelson and col. 23, lines 15 – 31 of Jones] wherein each of the shared resources is assigned to one of the scheduling domains and is not subject to domain migration [Section 3.2 Variable Partitioning, p. 8 of Feitelson]; and changing association of a first process of the plurality of processes from a first scheduling domain to a second scheduling domain, if the first process requires a resource associated with the second scheduling domain [thread migration; p. 7, 3rd paragraph of Feitelson].

9. As to claim 34, Feitelson as modified teaches a method implemented in a multiprocessor system, the method comprising:

executing a software program that defines a plurality of tasks [pp. 6 – 7, Section 3.1 of Feitelson and col. 24, lines 15 – 52 of Jones] and assigns each of the plurality of tasks to one of a plurality of scheduling domains [pp. 9 – 10 of Feitelson], wherein none of the plurality of scheduling domains is bound to any one processor in the system [Section 3.3, p. 9 of Feitelson];

running a plurality of processes, each of the plurality of processes performing a different one of the plurality of tasks [pp. 6 – 7, Section 3.1 of Feitelson and col. 24, lines 15 – 52 of Jones], wherein each of the plurality of processes is run by a processor independent of an identity of the processor [Section 3.1, p. 7 of Feitelson];

prohibiting concurrently executing processes performing tasks that are each assigned to a same scheduling domain [Section 3.2, p. 8 of Feitelson and col. 28, lines 6 – 25 of Jones];

allowing concurrently executing processes performing tasks that are each assigned to a different one of the plurality of scheduling domains [Section 3.2, p. 8 of Feitelson and col. 23, lines 15 – 31 of Jones]; and allowing changing assignment of at least one task from a first scheduling domain to a second scheduling domain during executing the software program, if the at least one task requires a resource assigned to the second scheduling domain [thread migration; p. 7, 3rd paragraph of Feitelson], wherein each of the shared resources is not subject to domain migration [Section 3.2 Variable Partitioning, p. 8 of Feitelson].

10. As to claim 35, Feitelson as modified teaches a processing system comprising:

a plurality of processors [pp. 6 – 7, Section 3.1 of Feitelson and col. 7, line 63 – col. 8, line 13 of Jones], each processor having an identity [col. 22, lines 28 – 38 of Jones];

a memory coupled to each of the plurality of processors [col. 7, lines 32 – 63 of Jones], the memory storing instructions which, when executed by one or more of the plurality of processors, cause the one or more of the plurality of processors to perform a method comprising:

executing a software program associating a plurality of tasks [p. 5, Section 2.3 and p. 6, Section 3 of Feitelson] with a plurality of scheduling domains [Section 3.1, p. 7 of Feitelson] and assigning a plurality of resources to the plurality of scheduling domains [Section 3.2 Variable Partitioning, p. 8 of Feitelson], wherein none of the plurality of scheduling domains is bound to any one processor of the plurality of processors [Section 3.3, p. 9 of Feitelson], and wherein each task of the plurality of tasks in scheduled on one of the processors independent of the identity of the processor [Section 3.1, p. 7 of Feitelson];

prohibiting concurrently executing processes to perform tasks that are each associated with a same scheduling domain [Section 3.2, p. 8 of Feitelson and col. 28, lines 6 – 25 of Jones] but allowing concurrently executing processes to perform tasks that are each associated with a different one of the plurality of scheduling domains [Section 3.2, p. 8 of Feitelson and col. 23, lines 15 – 31 of Jones]; and

changing association of a first task of the plurality of tasks from a first scheduling domain to a second scheduling domain, if a process performing the first task requires a resource assigned to the second scheduling domain [p. 7, 3rd paragraph of Feitelson], wherein the resource is not subject to domain migration [Section 3.2 Variable Partitioning, p. 8 of Feitelson].

11. As to claim 36, Feitelson as modified teaches a computer-readable storage medium storing instructions therein which, when executed by one or more processors of a processing system, cause the one or more processors to perform a method comprising:

executing a software program that defines a plurality of tasks [pp. 6 – 7, Section 3.1 of Feitelson and col. 24, lines 15 – 52 of Jones] and assigns each of the plurality of tasks to one of a plurality of scheduling domains [pp. 9 – 10 of Feitelson], wherein none of the plurality of scheduling domains is bound to any one processor in the system [Section 3.3, p. 9 of Feitelson];

running a plurality of processes, each of the plurality of processes performing a different one of the plurality of tasks [pp. 6 – 7, Section 3.1 of Feitelson and col. 24, lines 15 – 52 of Jones], wherein each of the processes is run on a processor independent of the identity of the processor [Section 3.1, p. 7 of Feitelson];

prohibiting concurrently executing processes performing tasks that are each assigned to a same scheduling domain [Section 3.2, p. 8 of Feitelson and col. 28, lines 6 – 25 of Jones];

allowing concurrently executing processes performing tasks that are each assigned to a different one of the plurality of scheduling domains [Section 3.2, p. 8 of Feitelson and col. 23, lines 15 – 31 of Jones]; and

allowing changing assignment of at least one task from a first scheduling domain to a second scheduling domain during executing the software program, if the at least one task requires a resource assigned to the second scheduling domain [p. 7, 3rd paragraph of Feitelson], wherein each of the shared resources is not subject to domain migration [Section 3.2 Variable Partitioning, p. 8 of Feitelson].

12. As to claim 37, Feitelson as modified teaches a method, comprising:

associating a task of a plurality of tasks [pp. 6 – 7, Section 3.1 of Feitelson and col. 24, lines 15 – 52 of Jones] with a scheduling domain of a plurality of scheduling domains [p. 5, Section 2.3 and p. 6, Section 3 of Feitelson], wherein the plurality of tasks share one or more resources [col. 28, lines 6 – 24 of Jones] and each of the one or more resources is assigned to one of the plurality of scheduling domains [Section 3.2 Variable Partitioning, p. 8 of Feitelson];

scheduling the task, using the scheduling domain [processors; pp. 6 – 7, Section 3.1 of Feitelson and col. 7, line 63 – col. 8, line 13 of Jones], on a processor in a multiprocessor system that includes a plurality of processors [pp. 6 – 7, Section 3.1 of Feitelson and col. 7, line 63 – col. 8, line 13 of Jones], independent of an identity of the processor [Section 3.1, p. 7 of Feitelson];

prohibiting tasks that are each associated with a same scheduling domain from running concurrently [Section 3.2, p. 8 of Feitelson and col. 28, lines 6 – 25 of Jones];

allowing tasks that are each associated with different scheduling domains to run concurrently [Section 3.2, p. 8 of Feitelson and col. 23, lines 15 – 31 of Jones]; and

changing association of the task from a first scheduling domain to a second scheduling domain, if the task requires a resource assigned to the second scheduling domain [p. 7, 3rd paragraph of Feitelson], wherein each of the shared resources is not subject to domain migration [Section 3.2 Variable Partitioning, p. 8 of Feitelson].

13. As to claim 14, Feitelson teaches at least one of the set of tasks is associated with more than one scheduling domain of the plurality of scheduling domains [Section 3.2, p. 8 of Feitelson].

14. As to claim 16, Feitelson as modified teaches a scheduler includes a plurality of runnable queues one per scheduling domains [col. 20, line 50 – col. 21, line 29 of Jones].

15. As to claim 31, Feitelson as modified teaches allowing concurrently executing processes that are not associated with any one of the plurality of scheduling domains [col. 28, lines 6 – 25 of Jones].

16. As to claim 32, Feitelson teaches at least one of the plurality of processes is associated with more than one of the plurality of scheduling domains [Section 3.2, p. 8 of Feitelson].

17. As to claim 33, Feitelson as modified teaches a plurality of scheduling domains is associated with a different one of a plurality of runnable queues [col. 20, line 50 – col. 21, line 29 of Jones].

CONTACT INFORMATION

18. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Li B. Zhen whose telephone number is (571) 272-3768. The examiner can normally be reached on Mon - Fri, 8:30am - 5pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571)272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Li B. Zhen/
Primary Examiner, Art Unit 2194

Li B. Zhen
Primary Examiner
Art Unit 2194